NGÔN NGỮ MÔ HÌNH HOÁ THỐNG NHẤT UML VÀ NGÔN NGỮ ĐẶC TẢ SDL

Trịnh Thị Thuý Giang

Trường Đại học Khoa học Tự nhiên

Một câu nói mà những nhà thiết kế phần mềm luôn tâm niệm đó là: "*Một bức tranh bằng trăm nghìn lời nói*", điều này giống như việc mô hình hoá. ở đây tôi xin được trình bày về ngôn ngữ mô hình hoá thống nhất UML (Unified Modeling Language), một ngôn ngữ mà ngày nay đang được sử dụng rất nhiều trong công nghiệp phần mềm và so sánh nó với một ngôn ngữ đặc tả trước đó là ngôn ngữ SDL (Specification and Description Language).

1. NGÔN NGỮ UML (UNIFIED MODELING LANGUAGE)

UML là ngôn ngữ mô hình để thực hiện mọi chức năng của hoạt động làm phần mềm: *Làm trực quan hoá* (visualizing), *đặc tả* (specifying), *xây dựng* (constructing), *làm tài liệu* (documenting) cho thiết kế phần mềm [2]. Như vậy, UML về cơ bản định nghĩa một siêu mô hình và không mô tả "kỹ thuật phát triển".

UML trợ giúp cho cả 5 giai đoạn trong phát triển hệ thống: *Phân tích yêu cầu, Phân tích, Thiết kế, Lập trình, Kiểm thử.*

Một lợi thế rất lớn trong UML là nó có các View chỉ ra các khía cạnh khác nhau của hệ thống được mô hình. Một view không phải là một mô tả đồ họa mà nó bao gồm một số các biểu đồ trừu tượng. Tổng hợp tất cả các View cho chúng ta một cái nhìn tổng quan về hệ thống. Trong UML có 5 loại View: use case view, design view, process view, implementation view, deployment view [3].

Uses case view bao gồm các use case mô tả hành vi của hệ thống được xem bởi các user của nó, phân tích và kiểm thử. View này không

chỉ rõ tổ chức của một hệ thống phần mềm. Trong UML khía cạnh tĩnh của View này thể hiện trong các biểu đồ use case còn khía cạnh động thể hiện trong các biểu đồ tương tác, biểu đồ statechart và các biểu đồ hoạt động.

- Design view bao gồm các lớp, các giao diện và các cộng tác. View này chủ yếu trợ giúp cho các yêu cầu chức năng của hệ thống, có nghĩa là các dịch vụ mà hệ thống sẽ cung cấp cho các user của nó. Với UML mặt tĩnh của View này thể hiện trọng các biểu đồ lớp và biểu đồ đối tượng; mặt động thể hiện trong các biểu đồ tương tác, biểu đồ trạng thái và biểu đồ hoạt động.

- Process view bao gồm các dòng (thread) và các tiến trình (process), trong đó có các cơ chế tương tranh và động bộ của hệ thống. View này chủ yếu nhằm vào sự thực hiện, khả năng và các tiến trình của hệ thống.

- Implementation view bao gồm các thành phần và các file mà được sử dụng để tập hợp và đưa ra hệ thống vật lý. View này chủ yếu tập trung vào việc quản lý cấu hình của hệ thống đưa ra.

- Deployment view bao gồm các node mô tả yêu cầu phần cứng khi hệ thống thực hiện. Nó chỉ ra việc triển khai hệ thống trong 1 kiến trúc vật lý có dùng máy tính và các thiết bị (các nút). Nói cách khác là nó mô tả việc triển khai vật lý của hệ thống. View này dành cho người phát triển, tích hợp và kiểm thử.

Một mô hình trong UML bao gồm một số các VIEW mô tả các khía cạnh khác nhau của hệ thống. Chỉ cần kết hợp các VIEW chúng ta có thể tạo được một bức tranh toàn cảnh của hệ thống. Một VIEW không phải là đồ thị, các nội dung của nó là được mô tả bằng các biểu đồ kết hợp (combining) các phần tử mô hình: Lớp, đối tượng, nút, thành phần và các mối quan hệ [4].

2. NGÔN NGỮ ĐẶC TẢ SDL CHUẨN CỦA ISO

SDL (Specification and Description Language) được xây dựng dựa trên cơ chế nhãn dịch chuyển; các công cụ của SDL bắt buộc cả người sử dụng và người thiết kế trở lên hình thức hơn. SDL phù hợp cho việc

nhận biết các lỗi và khả năng làm cho các mô hình trở lên sống động và giúp cho việc trả lời câu hỏi : "What if?" [5].

SDL được sử dụng khá rộng rãi trong truyền thông công cộng và nó trợ giúp rất tốt cho các hệ thống này nhờ bộ công cụ đa dạng, ngoài ra SDL còn có nhiều giá trị về mặt thương mại. Mặc dù tên SDL ngụ ý sử dụng cho đặc tả và mô tả, nhưng SDL ngày này đã được sử dụng rộng rãi cho các mục tiêu khác như là thiết kế, lập trình ở mức cao với nhiệm vụ phức tạp hoặc phân tán như: Truyền thông vệ tinh, hàng không, thiết bị y tế, hệ thống điều khiển xe lửa, các giao thức truyền thông trong các ô tô,...

SDL có hai cách mô tả: Mô tả đồ hoạ được gọi là GR (Graphical Representation) và mô tả văn bản gọi là PR (Phrase Representation). SDL cho phép chuyển đổi tự động và thay thế lẫn nhau giữa PR và GR.

Một hệ thống SDL bao gồm các thành phần sau:

- *Cấu trúc* (structure): system, block, process, và các thủ tục.

- *Sự truyền thông* (communication): các tín hiệu với các tham số tín hiệu lựa chọn và các kênh (channel) hoặc các đường truyền tín hiệu (signal route).

- *Hành vi* (behavior): các tiến trình (process)

- *Dữ liệu* (data): các dạng dữ liệu trừu tượng (ADTs)

- *Sự kế thừa* (Inheritance): mô tả các mối quan hệ và đặc tả.

3. SO SÁNH GIỮA UML VÀ SDL

Để so sánh giữa UML và SDL ở đây tôi tập trung vào các vấn đề chính sau:

Giống nhau:

- Các hành vi (cấu trúc động): UML và SDL đều dựa trên cơ chế ***trạng thái nhãn dịch chuyển***. UML có biểu đồ trạng thái, SDL có máy trạng thái hữu hạn.

- Các cấu trúc tĩnh: UML có các hệ thống con, các lớp và các liên kết; SDL có các khối, các tiến trình và các kênh.

- UML có biểu đồ tuần tự; SDL có MSC (Message Sequence Chart)

Khác nhau:

- Trong SDL, khái niệm ***hệ thống là trung tâm***; Trong UML ***không có khái niệm hệ thống***. Chỉ có các lớp và các liên kết giữa chúng. Cấu trúc đối tượng có thể được mô tả nhưng chỉ với mục tiêu là rõ. UML có 5 view thể hiện các khía cạnh khác nhau của hệ thống, tạo nên một bức tranh toàn cảnh của hệ thống cần xây dựng, do vậy không có khái niệm hệ thống tồn tại trong UML. Có thể coi SDL chỉ có 1 view, khi xây dựng hệ thống với SDL nó nhìn toàn bộ hệ thống một cách bao quát trên một khía cạnh nào đó.

- UML ***có tính mở*** với nghĩa: UML có thể mở rộng cho mọi công ty có thể xây dựng các bộ công cụ riêng cho mình dựa trên các khái niệm của UML. Tuỳ thuộc vào yêu cầu của mình họ có thể xây dựng một ngôn ngữ tuân theo các khái niệm của UML. Với SDL thì ***tính mở yếu*** hơn, nó không được dùng phổ biến để xây dựng cho mọi hệ thống phần mềm, mà phần lớn nó chỉ được áp dụng trong các hệ thống phân tán, các hệ thống truyền thông, hệ thống thời gian thực.

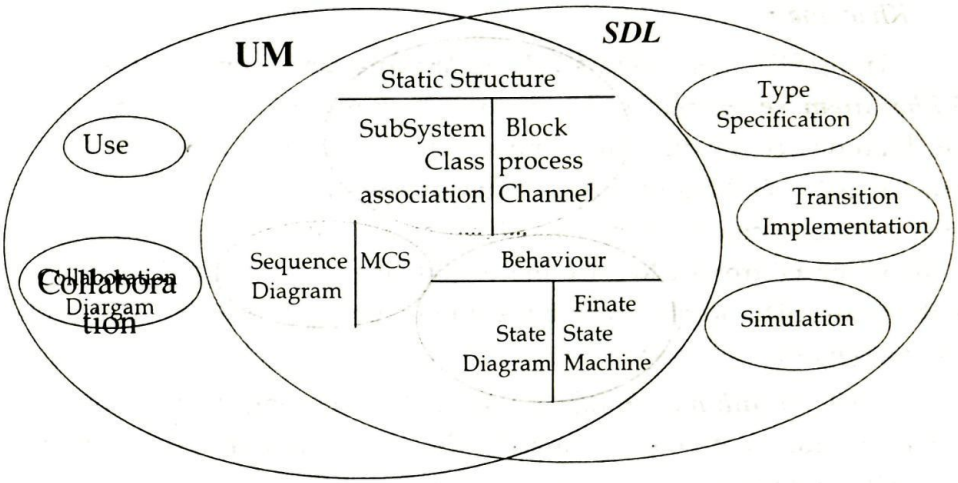
- Trong SDL, một tác nhân có thể được làm từ một cấu trúc phức tạp của các tác nhân khác và từ đó có thể định nghĩa một dạng tác nhân ghép. Trong trường hợp này, một ***kiểu ghép*** có thể được định nghĩa và sử dụng như một phần tử với giao diện định nghĩa tốt. UML thiếu một khái niệm tương ứng với các kiểu ghép. Sự ghép này là một liên kết đặc biệt giữa các lớp trong UML.

- Trong SDL, sự tương tranh và truyền thông không đồng bộ giữa các tác nhân là một ***quy tắc***. Các máy trạng thái định nghĩa dãy các hành vi của các tác nhân. Trong UML, tương tranh và truyền thông không đồng bộ là một ***sự lựa chọn***.

- UML có các use - case dùng để nắm bắt các yêu cầu, các biểu đồ cộng tác mà SDL không có các thành phần này.

- SDL có các kiểu đặc tả, các sự chuyển dịch có thực hiện trong dạng chi tiết đầy đủ, trong UML không trợ giúp cho các quá trình này.

Hình sau chỉ ra phần chung, các biểu đồ và các khái niệm có thể dùng được trong UML và SDL. Chúng chia sẻ sự đặc tả cho cấu trúc tĩnh, hành vi và các kịch bản (scenario).



So sách các đặc trưng giữa UML và SDL

4. PHÁT TRIỂN PHẦN MỀM DỰA TRÊN UML VÀ SDL

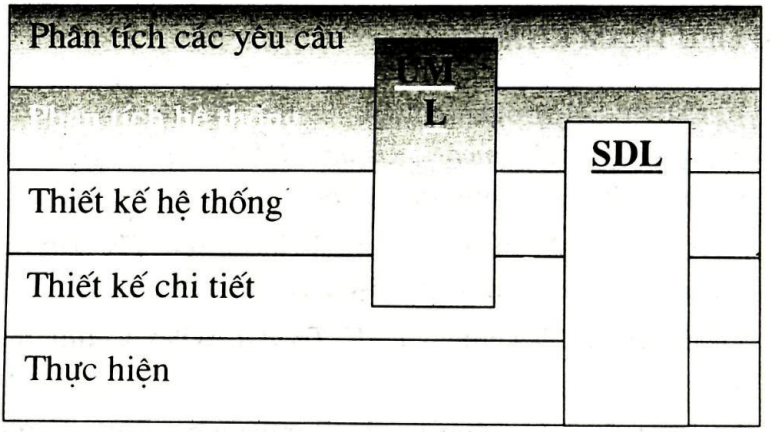
Việc sử dụng UML có nghĩa là chúng ta xây dựng các đặc tả trong UML và thực hiện trong các ngôn ngữ lập trình hướng đối tượng như C^{++} , Java,..... Như vậy UML có các khái niệm cho phép xây dựng các bộ công cụ để có thể tự động dịch các mô hình đặc tả ra mã chương trình. Tuy nhiên đối với các hệ thống thời gian thực việc sử dụng UML còn gặp nhiều vấn đề không thuận lợi. Cụ thể là đối với các hệ thống lớn, phức tạp trong truyền thông. SDL lại có ưu thế trong vấn đề này.

Để phát triển các tiến trình phần mềm thông thường chúng ta bắt đầu từ những use case đơn giản, vì như vậy chúng ta dễ dàng nắm bắt được hệ thống của chúng ta thông qua các use case; sau đó chuyển các use case này vào các biểu đồ SDL. Như vậy để thực hiện các tiến trình phần mềm chúng ta sẽ bắt đầu với UML và sau đó chuyển đến SDL cho các giai đoạn sau của tiến trình phần mềm. UML trợ giúp rất mạnh cho các giai đoạn sớm của quá trình phát triển phần mềm còn SDL có nhiều thuận lợi trong giai đoạn thiết kế và giai đoạn thực hiện sau đó.

Trong giai đoạn thiết kế, UML được sử dụng rất tốt, trong giai đoạn này chúng ta sử dụng các use case và các biểu đồ tuần tự để mô tả yêu cầu và các hoạt động của hệ thống. Hơn nữa các biểu đồ lớp không

hình thức có thể được dùng cho việc mô tả đồ hoạ cho các vấn đề đặc biệt. Sau khi hoàn thành các bước này thì chúng ta đã thực hiện xong việc đặc tả yêu cầu (R-Spec). Các biểu đồ triển khai cung cấp một cái nhìn khái quát toàn bộ hệ thống trong giai đoạn thiết kế. Sau đó các biểu đồ trạng thái và các biểu đồ tuần tự có thể dịch trực tiếp ra mã chương trình thông qua SDL. Điều này có nghĩa rằng, kết quả thu được trong giai đoạn thiết kế không chỉ có giá trị cung cấp tài liệu cho đặc tả chức năng và đặc tả dữ liệu cũng như với đặc tả yêu cầu mà kết quả này còn được sử dụng để trực tiếp dịch ra mã chương trình [1].

Do SDL và UML có các khái niệm tương tự nhau và dựa vào những so sánh trên chúng ta thấy rằng việc kết hợp giữa UML và SDL sẽ cho chúng ta một ngôn ngữ đặc tả rất tốt cho quá trình phát triển phần mềm.



Sự tương quan giữa UML và SDL trong các giai đoạn phát triển phần mềm

5. KẾT LUẬN

UML là một ngôn ngữ mô hình hoá sử dụng các mô tả đồ hoạ dùng cho sự hình dung, sự đặc tả, sự xây dựng và làm tài liệu, đây chính là những công cụ rất mạnh cho các hệ thống phần mềm lớn. UML được sử dụng khá phổ biến và nó đã được ứng dụng thành công để xây dựng các hệ thống trong: thương mại điện tử, các hệ thống chỉ huy và điều khiển, các trò chơi máy tính, điện tử trong y học, ngân hàng, bảo hiểm, điện thoại, robot và khoa học điện tử áp dụng trong hàng không. UML

đã vượt qua hầu hết các ngôn ngữ lập trình truyền thống, nó có các ánh xạ chuyển từ UML đến Java, C++, Smalltalk, Visual Basic, Ada,....

SDL là một ngôn ngữ chuẩn của ISO (International Organization for Standard) nó dùng để đặc tả, để xác minh, phân tích, thực hiện, kiểm thử và thao tác. SDL có thể thực hiện được những điều khiển phức tạp của các hệ thống truyền thông và OSI (Open systems International).

Bài báo đã nêu bật được những ưu điểm và nhược điểm khi sử dụng hai ngôn ngữ trên cho quá trình phát triển phần mềm và so sánh chúng. Đề tài đã được tiến hành tiếp tục với phần biến đổi từ UML sang SDL cũng như từ SDL sang UML giúp cho các nhà phát triển phần mềm có được sự linh động trong đặc tả phần mềm, xin phép sẽ được trao đổi cùng quý vị quan tâm trong một dịp khác.

TÀI LIỆU THAM KHẢO

1. Christian Schwingenschlửgl, Stefan Schửnauer. *Application and Service Depelopment Using UML and SDL.* Institute of Communication Networks, 1999.
2. G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language User Guide.* Addison-Wesley, 1998.
3. H. Eriksson, M. Penker. *UML Toolkit.* Wiley Computer Publishing, 1998
4. I. Jacobson, G. Booch, J. Rumbaugh. *The Unified Software Development Process.* Addison-Wesley, 1999.
5. Kenneth J.Turner, *Using Formal Description Techniques.* JOHN WILEY & SONS, 1993.